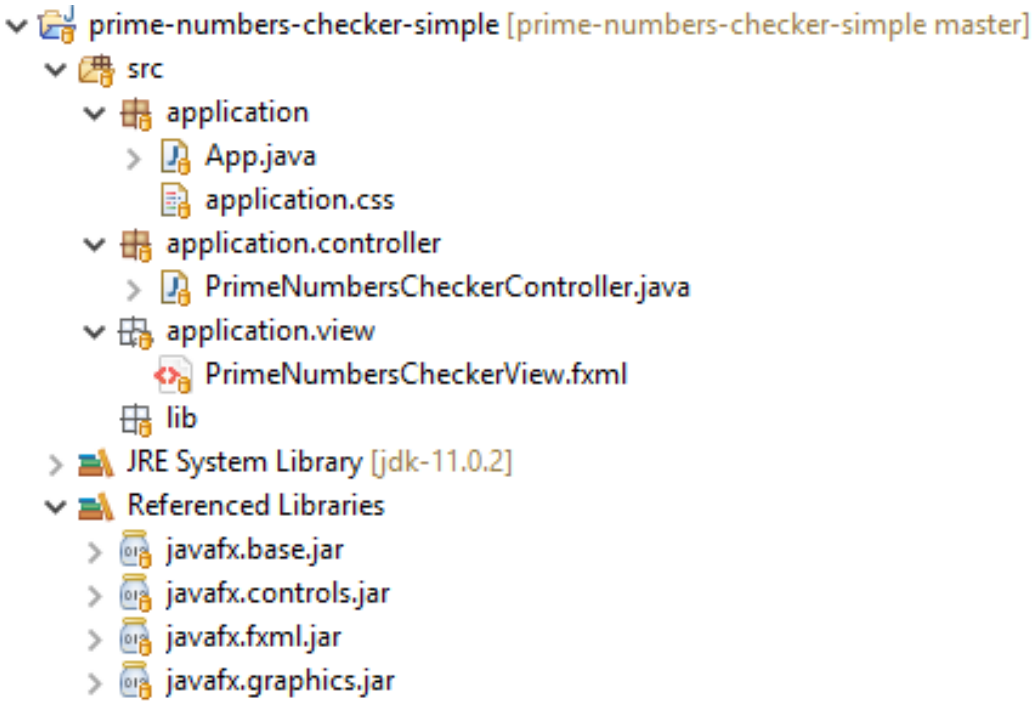
JavaFX mit SceneBuilder
Primzahlen-Checker mit FXML - Demo 1

Ausgangslage:

- Eine Skizze erstellt, aus der ersichtlich ist, wie das GUI ungefähr aussehe sollte
- Klassen und FXML-Files:
  - Im Paket `application` ist die Klasse erstellt, in der die Methode `start` enthalten ist (die Klasse `App`)
  - Im Paket `application.controller ist` die Klasse enthalten, in der das Verhalten implementiert ist (`PrimeNumberController`)
  - Im Paket `application.view` sind die FXML-Datei enthalten, welche das GUI (View) deklarativ beschreiben
- Im `lib`-Verzeichnis sind die JavaFX Module, die benötigt werden abgelegt und dem "Build Path" hinzugefügt
- Hinweis:
  Bei einem Maven-Projekt werden die FXML-Dateien in das Verzeichnis `src\main\resources` abgelegt.

JavaFX mit SceneBuilder
Primzahlen-Checker mit FXML - Demo 2

- Das Projekt könnte in `eclipse` wie folgt aussehen:



- Die FXML-Datei `PrimeNumbersCheckerView.fxml` wird mit SceneBuilder geöffnet und editiert

JavaFX mit SceneBuilder
Primzahlen-Checker mit FXML - Demo 3

- Im Accordion-Panel Hiearchy ist nur die Container-Komponente vom Typ AnchorPane ersichtlich, weitere Komponenten fehlen noch
- Damit die Container-Komponente sichtbar wird, geben wir unter `Inspector` → `Layout` die Werte für bevorzugte Breite und bevorzugte Höhe (300 und 120) ein
- Um das gewünschte Layout zu erhalten, werden nacheinander die restlichen Komponenten eingefügt
- Wichtig:
  - Jede Komponente, auf die Zugriff aus der Controller-Klasse möglich sein muss, muss einen Text-Wert für **`fx:id`** bekommen
  - Der **`fx:id`** Wert muss mit dem Namen des Feldes in der Controller Klasse, das mit `@FXML` annotiert wird, identisch sein!!!

JavaFX mit SceneBuilder
Primzahlen-Checker mit FXML - Demo 4

- Die erste Label-Komponente soll mit dem Text "`Enter an integer number:`" versehen werden (*Properties → Text*)

- Die TextField-Komponenten soll
  - unterhalb der ersten Label-Komponente positioniert werden
  - den Namen `txtNumber` erhalten (*Code → fx:id*)

- Die Button-Komponente soll
  - unterhalb der TextField-Komponente positioniert werden und
  - mit dem Text "`Check`" versehen werden (*Properties → Text*)

JavaFX mit SceneBuilder
Primzahlen-Checker mit FXML - Demo 5

- Es werden zwei weitere Label-Komponenten benötigt: eine für die Resultat- und eine für Fehler-Meldung

- Eine neue Label-Komponente
  - rechts von der TextField Komponente positionieren
  - mit einem Leerschlag-Text versehen (*Properties →Text*)
  - mit dem Namen lblResultat versehen (*Code → fx:id*)

- Eine weitere Label-Komponenten einfügen und
  - an die gleiche Position wie die vorherige Label-Komponente positionieren
  - mit einem Leerschlag-Text versehen (*Properties → Text*)
  - mit dem Namen lblError versehen (*Code → fx:id*)

- Hinweis:
  Der Text für die beiden Label-Komponenten wird während der Programmausführung gesetzt, je nach dem, ob die Ergebnis- oder die Fehler-Meldung angezeigt werden soll (darum die obligatorische `fx:id` Angabe)

JavaFX mit SceneBuilder
Primzahlen-Checker mit FXML - Demo 6

- Die Text-Farbe der `lblError`-Komponente soll rot sein, um besser aufzufallen

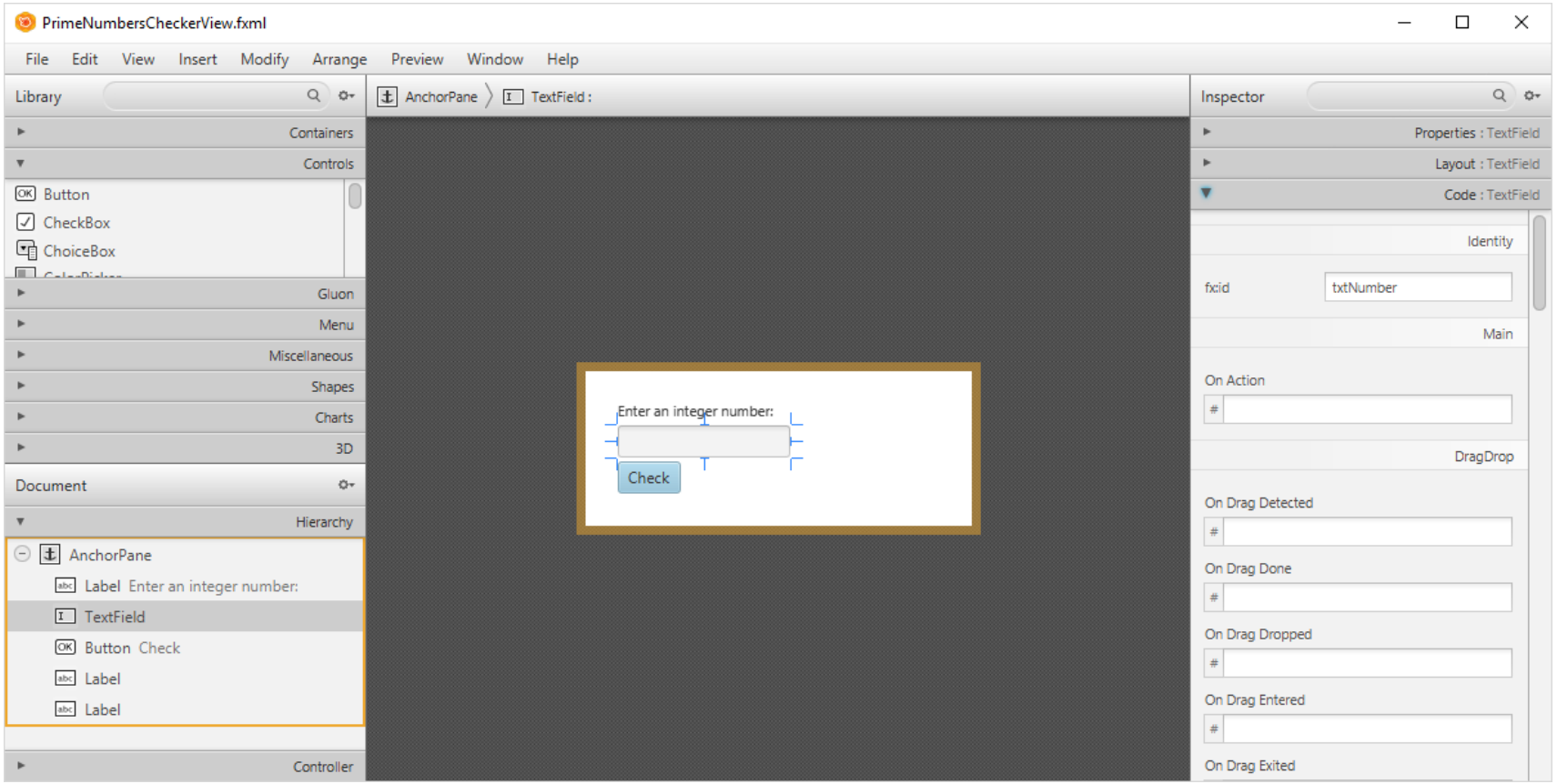
- Textfarbe wie folgt einstellen:
  - Die `lblError` Komponente im Accordion-Panel *Hierarchy* auswählen
  - Unter *Properties* → *Text Fill* rote Farbe (`#ff0000`) auswählen

- Falls gewünscht, kann auch unter *Properties* → *Font* eine andere Schrift (Schrift-Familie, Grösse etc.) eingestellt werden

JavaFX mit SceneBuilder

Primzahlen-Checker mit FXML - Demo 7

Das erstellte GUI könnte in SceneBuilder wie folgt aussehen:



JavaFX mit SceneBuilder
Primzahlen-Checker mit FXML - Demo 8

Die Controller-Klasse `PrimeNumbersCheckerController`:

```
package application.controller;

import javafx.fxml.FXML;
import javafx.scene.control.*;

public class PrimeNumbersCheckerController {
        @FXML
        private TextField txtNumber;
        @FXML
        private Label lblResult;
        @FXML
        private Label lblError;
        @FXML
        private void check() {
           // TODO ...
        }
}
```

JavaFX mit SceneBuilder
Primzahlen-Checker mit FXML - Demo 9

- Die Controller-Klasse ist implementiert
- Das GUI ist in der FXML-Datei implementiert
- Es fehlt:
  - Die Angabe der Controller-Klasse in der FXML-Datei
  - Die Angabe, welche Methode ausgeführt werden soll, wenn auf die Schaltfläche geklickt wird
- Die Angabe der Controller-Klasse und der Action-Methode kann sowohl mit Hilfe von SceneBuilder als auch direkt in der FXML-Datei vorgenommen werden

JavaFX mit SceneBuilder
Primzahlen-Checker mit FXML - Demo 10

- Um die Controller-Klasse und die Action-Methode mit SceneBuilder anzugeben, muss die FXML-Datei mit SceneBuilder geöffnet werden
- Angabe der Controller-Klasse:
  - Root-Container markieren
  - Im Accordion-Panel *Controller* im *Controller class* Kombinationstfeld die Klasse `application.controller.PrimeNumbersCheckerController` wählen
- Angabe der Action-Methode:
  - Schaltfläche mit Text "Check" markieren
  - Unter *Code → onAction* die Methode `check` wählen
- Schaltfläche als default-Button setzen (*Properties → Default Button*)
- Änderungen speichern, damit die FXML-Datei updatet wird

JavaFX mit SceneBuilder
Primzahlen-Checker mit FXML – Demo 11

- Abschluss:
  - GUI ist deklarativ beschrieben
  - Verhalten ist in der Controller-Klasse beschrieben

- Es fehlt noch die Implementierung der Methode start:
  - stage-Titel setzen
  - FXML-Klasse laden
- etc.

JavaFX mit SceneBuilder

Primzahlen-Checker mit FXML – Demo 12

Die Methode `start`:

```
@Override
public void start(Stage stage) throws Exception {

   try {
      stage.setTitle("Prime-Checker");

      AnchorPane root = FXMLLoader.load(getClass()
      .getResource("view/PrimeNumbersCheckerView.fxml"));

      Scene scene = new Scene(root, 400, 120);
      scene.getStylesheets().add(getClass()
      .getResource("application.css").toExternalForm());

      stage.setScene(scene);
      stage.show();
   } catch (Exception e) {
      System.out.println("Error!!!\n");
   }
}
```